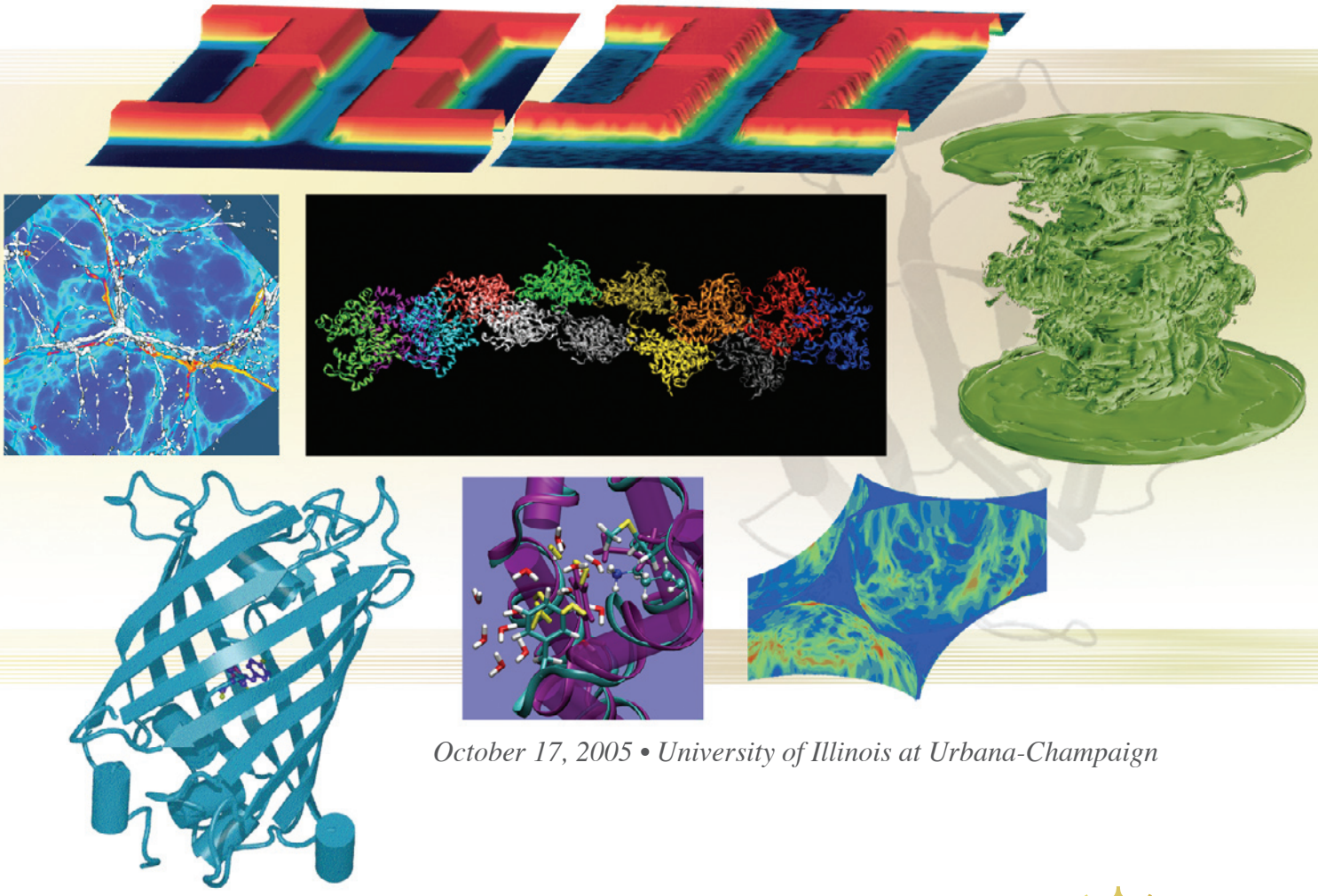
# Report of the High Performance Computing Town Hall Meeting:
## Science, Requirements, and Benchmarks



*October 17, 2005 • University of Illinois at Urbana-Champaign*

NSF

# Report of the High Performance Computing Town Hall Meeting

October 17, 2005
Held at the University of Illinois at Urbana-Champaign
Report date: April 13, 2006

## Contents



## Authors of the Report


- David Ceperley, University of Illinois Urbana-Champaign (Chair)
- Paul Fischer, Argonne National Laboratory
- Steven Gottlieb, Indiana University
- Robert Harrison, University of Tennessee, ORNL
- Luis Lehner, Louisiana State University
- Roy Williams, California Institute of Technology

## Section 1: Goals of the Town Hall Meeting

As summarized in "A Science-based Case for Large Scale Simulation" (SCALES) report (see below), computing and data handling resources need to be increased by 100-fold within 5-10 years so that, in combination with concurrent advances in theoretical and computational methods, the expanded capabilities will allow for scientific breakthroughs. Without such advances, the nation will fail to realize the scientific discoveries that the entire infrastructure is intended to enable.

The NSF has been working rapidly to develop and execute an implementation plan for cyberinfrastructure across the Foundation. High-performance computing (HPC) is an important component of cyberinfrastructure. Solicitations are planned to acquire high-performance computing systems on an ongoing basis in the next few years. The Town Hall Meeting provided an opportunity for the NSF Mathematical and Physical Sciences (MPS) community to discuss important issues related to high performance computing and to inform NSF of its views.

Technological barriers to research include applied and numerical math, algorithm, software and hardware. This meeting discussed the exciting cyberscience that specifically requires high-performance computing and the scientific opportunities opened by HPC. To help guide NSF investments and to help the community decide what resources are best suited for the scientific problems studied by the MPS community, the Town Hall Meeting focused on the issue of how to measure in a meaningful way the performance of a high performance computing system on MPS cyberscience problems. This report summarizes the Town Hall Meeting, and conclusions reached, particularly with respect to benchmarking.

Presentations on driving science, the codes that do it, and how they stress high-performance computing, sensible measures of performance, and good candidate benchmarks were addressed at the Town Hall Meeting. After the meeting, participants utilized a wiki (http://www.mcc.uiuc.edu/nsfhpc/) for continued discussions between assigned working groups to distill candidate benchmarks.

## Section 2: Computational/Scientific Challenges involving High-Performance Computing

Numerous presentations and previous reports have detailed some of the shortterm goals of high-performance computational science. Here we highlight a few of the challenges.

Computational methods have reached the point where they constitute an essential component of materials research. The combination of advances in theoretical concepts, computational methods and hardware capabilities is making possible simulations that, based on first principles, can predict the properties of new materials and reveal new insights. This is especially important for modern complex materials, where simulations can work together with experimental studies to discover new phenomena and materials systems here-to-fore unknown. The possibility of "materials by design", (this refers to the search for a material tailor-made to have desired properties) is a goal that many in the condensed-matter, and related communities will be working to achieve in the next decade. There is rapid progress on the correlated electron problem through the development of new algorithms such as quantum Monte Carlo and dynamical mean field theory. The goal of achieving the needed "chemical accuracy" is reachable for many important systems in the near future. A third direction is the development of multiple length and time scale simulations, the ability to go from a quantum electron description to compute macroscopic properties and dynamics. All of these research directions will require computer performance far in excess of existing resources.

Chemistry is concerned with the composition, structure and transformation of matter which includes the study of energetics and dynamics. The grand challenge in computational chemistry is to understand chemical transformation in sufficient detail to enable the control and *de novo* design of new molecules and processes. In this respect, chemistry is the central science and is tightly woven with the biological and material sciences, for instance in the study of protein structure and function, or in the pursuit of practical nano-scale devices. A particularly grand challenge

is the rational design of chemical catalysts from fundamental principles that promises new processes with improved activity and selectivity. A catalyst's role is to greatly improve the efficiency of a desired chemical reaction, and catalytic processes are directly involved in the synthesis of 20% of all industrial products. As identified in the recent SCALES report, a factor of 100-1000 increase in computational power promises to realize this ambition.

Within a few years, thanks to a new generation of extremely sensitive detectors, researchers will have an unprecedented amount of data which will be key to understanding our universe. The data analysis requires both inspecting the huge data-streams coming from the detectors and confronting them with theoretical models. This already requires massive computational power and the ability to access, distribute and manipulate tera- to petabytes of data. Data-handling capabilities are also urgent for those large simulations that store every time step: indeed I/O becomes a limiting factor. In these cases we expect large, synchronized parallel write operations, where each processor outputs part of a large file. Further along the cyberinfrastructure pipeline is the ability to move such large datasets to a collection place, and to analyze, visualize, and share these multi-terabyte datasets. Confronting data with theoretical models require the availability of accurate simulations of target systems of interest. These include:

• cosmological simulations produced with viable models (which require the ability to resolve vast length scales and track the system of extremely long integration times),

• astrophysical scenarios thought to be at the core of spectacular phenomena like gamma ray bursts, active galactic nuclei, supernovae etc. (which require integrating involved PDEs, resolve varied length scales, incorporate complicated physical mechanisms, etc), and

• systems that are likely to produce strong gravitational wave outputs (containing compact objects undergoing highly dynamical phases which in turn also require dealing with highly complicated PDEs, varied different length scales and very steep floating point operation count per point).

All these simulations are quite demanding of HPC as they require tera- and petaflop machines, with high speed connectivity and efficient I/O. Furthermore, this infrastructure must be efficiently maximized through smart algorithms ensuring high order accuracy, mesh adaptivity, etc. These simulations are necessary to decipher the data gathered, unravel the variety of phenomena and help open new views of our universe. HPC will be crucial in enabling these breakthroughs.

Quantum Chromodynamics (QCD) is the theory of the strong interaction of subatomic physics. Numerical studies of QCD (lattice QCD) address some of the most fundamental questions in high energy and nuclear physics, and are closely linked to the large experimental programs in these fields. They require very large scale calculations, which consume a significant fraction of the computational resources provided by the NSF Supercomputer Centers. Further progress will require calculations on more refined grids and work with new numerical approaches to the quarks. A straightforward extension of current calculations using improved staggered quarks would require approximately 200 Teraflop-years of computation. Other approaches may require even more power for comparably refined grids. If such calculations are not completed in a timely fashion, it will prevent us from learning the most that we can from the experiments that cost on the order of $100 million per year.

## Previous reports identifying major scientific challenges:


1. Identifying Major Scientific Challenges in the Mathematical and Physical Sciences and Their CyberInfrastructure Needs, held on April 21, 2004; http://www.nsf.gov/attachments/100811/public/CyberscienceFinal4.pdf
2. A Science-Based Case for Large-Scale Simulation; workshop held June 24-25, 2003; http://www.pnl.gov/scales/docs/volume1_72dpi.pdf; http://www.pnl.gov/scales/docs/SCaLeS_v2_draft_toc.pdf
3. Computation as a Tool for Discovery in Physics," October 2002, http://www.nsf.gov/pubs/2002/nsf02176/start.htm
4. Cyber Chemistry Workshop; workshop held October 3-5, 2004; http://bioeng.berkeley.edu/faculty/cyber_workshop
5. Materials Research Cyberscience enabled by Cyberinfrastructure; workshop held November 2004; http://www.nsf.gov/mps/dmr/csci.pdf
6. Multiscale Mathematics Initiative: A Roadmap; workshops held May 3-5, July 20-22, September 21-23, 2004; http://www.sc.doe.gov/ascr/mics/amr/Multiscale%20Math%20Workshop%203-%20Report latest edition.pdf
7. "Physics of the Universe," February 2004, http://www.ostp.gov/html/physicsoftheuniverse2.pdf
8. President's Information Technology Advisory Committee Report, August 16, 2005; http://www.nitrd.gov/pitac/reports/20050609_computational/computational.pdf

## Section 3: Summary of General Recommendations of the Town Hall Meeting

In the most general sense, the best criterion for rating computer purchases should be to maximize the scientific output per dollar: the usefulness to the scientific community should be the foremost consideration. One is buying a resource, not a machine. Of course, in practice, this criterion is difficult to apply, since the future needs of the scientific community are not completely predictable; there are gradual trends in both the algorithms used on high-end computing and in the scientific fields using those resources. In fact, there is active feedback because algorithms and codes are developed, to some extent, motivated by existing or planned computational resources;.

Another general comment that follows is that for capacity computing, it is important to acquire mature systems allowing demonstrated production usage. The majority of the scientific community is unwilling to port their codes to machines that are not ready, or to one-of-a-kind environments, which often do not last more than a single acquisition cycle. Benchmarking for the system software, with a realistic mix of jobs, is the best way to ensure that the machines can be used by the majority of the community.

A general consideration, recognized by the discussion, is the tension between capability and capacity (at a given level of capability) computing. Some scientists work in a steady state mode, where computational jobs are fed into the resource at a steady rate. Those scientists would be best served by the system that can deliver the most useful operations (e.g. FLOPs or memory references) per year, assuming that the machine has adequate communication, IO, disk storage etc. A second groups of scientists push the hardware to the limit and requiring the largest memory, communication bandwidth, or number of processors; they are less interested in steady state production than the ability to do "hero" calculations, perhaps leading to breakthroughs of computational science. A third mode is that of scientists developing new algorithms and applications, critical for the long term health of the field). Testing and refining codes make take a year or more, at which point, they need a "burst" of resources lasting for several months to make timely comparison with experiments, for funding, or conferences. It is important for the NSF to determine to what extent each type of computing is needed, to ensure that each type can be appropriately supported.

All of this discussion underscores that flexibility should be built into the acquisition process. It is quite clear that buying the largest possible homogenous computer is not the optimal solution for the scientific community as a whole. A diversity of platforms, such as currently exists among the NSF supported computers, will continue to be required. This diversity can be achieved in several ways: for example, one solution is to solicit bids for a shared-memory machine one year, and a large capacity machine another year. Or, perhaps, the solicitation can encourage institutions to propose acquiring multiple platforms with different characteristics, in a single proposal. In any case, diversity of platforms is required and the NSF will need to carefully anticipate the required capacity for each architecture in order to optimize the acquisitions.

Finally, we want to comment that the wiki format did not yield useful discussion since there was no way to focus the discussion, and to separate out expert contributions from general observations. The process did not include experts in benchmarking. There are several groups that have done extensive research in benchmarking and related topics; in the long term, the NSF should try to capture its investment in this research; taking the basic research into practice. We recommend funding benchmarking for scientific applications as part of NSF's cyber-infrastructure programs. Benchmarking experts should examine the codes suggested here and determine in which way these suggestions challenge the computers on which they will be run. They should also establish how accurately the performance of other important codes not among those selected can be predicted from the selected benchmarks. Finally, it would be useful to have performance models that could predict performance of benchmark codes based on low level characteristics of the computers, such a memory bandwidth, cache organization, and interprocessor communication characteristics.

Expertise in code tuning can also be helpful for scientists who wish or need to port their codes to platforms that are expected to be more cost effective in the long run. The expense of this work can most easily be justified for the codes that are consuming, or are expected to consume, the most resources.

The town hall process was very useful for allowing

many issues to be brought to the table; however, it would be valuable to augment the qualitative input obtained there by more quantitative and scientific analysis of the needs of the community. This clearly requires additional time and effort by both the community and NSF personnel, but in view of the longterm nature and expense of the anticipated acquisitions, this will certainly be a good investment

**Distillation of wiki discussion**

Chemistry and condensed matter physics are both diverse subjects and the computational needs of the associated tools vary greatly. Classical molecular dynamics (MD), which provides atomistic models of material and biological systems, is characterized by small memory requirements (circa 512 MB/process) and the demand for low-latency, high-bandwidth communications. This maximizes scalability and minimizes the wall-clock time required for each time-step of the simulation. *Ab initio* (plane wave) molecular dynamics, which adds electronic detail to atomistic simulations, typically requires more memory than classical MD simulations. In addition to excellent point-to-point communication, those applications require good global communication performance (bisection bandwidth). Very large *ab initio* MD simulations, as well as band-structure calculations, can require a large aggregate memory (1-2 TB). Atomic orbital (AO) codes typically require significant local memory (circa 2 GB per process) for workspace. AO density functional calculations have similar communication requirements to plane-wave simulations. Very accurate calculations, such as coupled cluster methods, in addition have large (multiple TB) aggregate memory needs but the computation is dominated by matrix multiplication and does not require low-latency communication. Good communication bandwidth, both local and global, is still required. The necessary disk space grows with the memory used and the AO methods can often use extra memory to avoid recomputation of intermediate results.

The Town Hall Meeting discussion on the execution of benchmarks largely centered around the advantages of involving the code developers and even their user community in the evaluation effort. Having the application developers involved in the development of the detailed benchmark inputs would greatly aid in capturing the essential performance characteristics in a minimum set of runs. Their involvement in the benchmark analysis would also potentially help with interpretation of results, though most major supercomputing institutions already have in-house experts familiar with the major codes. It was also suggested that small configurations of perhaps a "short list" of possible systems be made available on the net for some period to enable testing by the user community and more extensive code tuning for the benchmarks. However, the meeting group largely felt that the effort required for a consistent evaluation or tuning of codes would be very significant and that it might be considered as an activity worthy of future funding in advance of any procurement.

## Section 4: Existing Benchmarking Suites

The Department of Energy (DOE) National Energy Research Scientific computing Center (NERSC) http://www.nersc.gov procures a new supercomputer about every three years and has established a well-founded benchmark suite that was discussed by McCurdy during the NSF workshop. Details are available in his presentation and at the NERSC website. The tests and applications have some overlap with those identified here, but ultimately reflect the DOE workload as well as the different procurement approach and machine requirements. Perhaps of most value to this community is the testing of system throughput and availability.

**Relationship to existing NSF benchmarking suite**

Briefly, the previous (2005) benchmarks were:

1. HPC Challenge
2. SPIOBENCH
3. WRF
4. OOCORE
5. GAMESS
6. MILC
7. PARATEC
8. HOMME

**Overall rules for running the benchmarks**

The 2005 benchmarks also touch upon the need to measure and to test overall system throughput. The performance of single applications on a dedicated machine is not fully representative of a production system under the load of diverse applications. Do multiple jobs interact negatively with each other? Can the file system handle multiple jobs starting or check-pointing at the same time? Is the batch scheduler robust and efficient? Managing a mix of large and small jobs is a challenge and without a good backfilling algorithm, especially on machines requiring spatial locality of allocated resources, the overall machine usage can either become very inefficient, or large jobs wait a very long time in the queue.

In the context of our brief to support the procurement by MPS of a machine in 2006/7, specifically with the intent of enabling next-generation, scalable applications, the committee concluded that the above suite needed to be extended and modified as follows:

• Employ a wider ranger of languages, specifically including C++ and mixed-language applications.

• Incorporate a scalable application from molecular biology which was not previously included and presents unique challenges.

• Refine the fundamental benchmarking to enable greater analysis of the expected system capability, throughput, and scalability, and to include increased productivity programming models that employ one-sided communication/remote memory access.

• Modify the IO-intensive benchmarks in anticipation of diverse solutions to IO in scalable systems and to make the results more readily interpreted.

• Provide greater coverage of critical algorithms including spectral, uniform and adaptively refined mesh methods, along with tree and particle codes.

• Anticipate trends in computer architecture, such as multi-core chips, that require effective use of both coarse and fine grain parallelism.

## Section 5: Recommendations for MPS benchmarks

In developing this benchmark suite we considered the roles of benchmarking in machine evaluation and procurement:

• To measure fundamental and standard machine performance parameters in order to provide a rational basis for comparing the expected performance of diverse machines and the application benchmark results.

• To maximize and to enable informed balancing of the capability and the cost-effectiveness of computer systems for the expected scientific workload.

• To demonstrate the correct functioning of both hardware and software with realistic applications and under representative system load.

There is a strong connection between the science objectives, benchmarks, and high-priority machine requirements. The correct execution of a benchmark demonstrates the existence of a scientific capability and the high probability that related methods/ algorithms will also be ported readily and will function satisfactorily. The development of this benchmark suite has been challenging due to the absence of clearly defined, science-motivated requirements. There was some discussion of using benchmarks to assess productivity such as the times required to implement correctly and subsequently to optimize the performance of a scientific application. However, this is still a topic for research, and, is in part, addressed here by the availability of standard and emerging next-generation parallel programming tools.

In order to maximize the productive lifetime of the procured system, as well as, to increase the productivity of developers of current and new scientific applications, we also emphasize the availability and good-performance of one-sided communication mechanisms that go beyond the standard message-passing model. Examples of such models include library-based tools such Global Arrays and language extensions such as UPC and Co-Array Fortran. The HPCS program is also exploring three completely new parallel programming languages (Chapel, X10, Fortress) which incorporate many new and old ideas for increased productivity. Such tools have been demonstrated to greatly simplify the development of parallel programs by bringing the program content closer to the physics and to increase scalability by introducing greater asynchrony into execution. Global Arrays is in the public domain, is already available or readily ported; its transport layer is also used by Co-Array Fortran. Nearly all scalable molecular chemistry codes also use Global Arrays (or a derivative thereof) and it is therefore a requirement.

We anticipate that any system purchased in 2006/7 will contain many tens of thousands of processors, and that a significant emphasis will be placed upon the performance of applications that can use a large fraction of such resources to explore new science frontiers. For these reasons, we have largely selected benchmark codes that can now scale efficiently into this regime, or can credibly do so in the near future. The vendors are unlikely to be able to assemble

### Summary table of benchmark codes

| Benchmark | Domain | Algorithm | Language | Library use | Scalability |
|---|---|---|---|---|---|
| NAMD | Molecular Bio | Molecular dynamics | C++ | CHARM++ | 32K |
| MILC | Lattice QCD | ConjGrad., MD | C | MPI | 4096 |
| FLASH | Cosmology | AMR | F77, C, Python | – | 1000+ |
| PARATEC | Solid State Physics | Plane wave DFT | F90 | MPI | 2000+ |
| GAMESS | Quantum Chemistry | Gaussian HF, DFT, CC | F77/C | SCALAPACK, BLAS, DDI | 1000 |
| GASOLINE | Cosmology | - | - | MPI, POSIX,SHMEM | 512+ |
| CACTUS-GR | General Relativity | PDE | C, F77, F90 | MPI | 512+ |
| Global Array | Parallel computation utilities | - | - | - | - |

the final machine for testing, so the codes used for benchmarking should be sufficiently well understood and produce adequate information to allow extrapolation of results to the proposed configuration. This was accounted for in the 2005 set of benchmarks. However, we must also accommodate the proposal of multiple architectures that collectively provide the requested capability while realizing much improved price performance. For instance, molecular dynamics and QCD codes typically do not require much memory or disk IO, and have a common requirement for low-latency, high-bandwidth communication. Some *ab initio* electronic structure codes require large local and aggregate memory and, with the large memory usage, comes a requirement for a commensurate disk IO bandwidth and capacity. Configuring either separate or subset systems to satisfy these needs can provide a tremendous return in both price performance and peak capability.

**CFD/GR Benchmarking Considerations:** There is no single code or even a handful of codes that dominate the high-performance computing segment of computational fluid dynamics (CFD). Most modern CFD codes, however, comprise a few common kernels that are central to performance. Local computations typically involve sparse matrix vector products, dense matrix-matrix products, or multidimensional fast Fourier transforms (FFT). Global operations typically involve either nearest neighbor exchanges (e.g., for finite volume, difference, or element methods) or matrix transposes (e.g., for global spectral methods). Generally, the central CFD kernels are scalable, provided that there are enough points per processor. For example, from measured computation and communication performances for distributed memory architectures over the past two decades, one can estimate that roughly 1000-10,000 points per processor will suffice for roughly 50% parallel efficiency. (In the case of global spectral methods, one also requires sufficient bisectional bandwidth in the network to support the requisite data transposes.)

As one migrates from one Petascale platform to another, a question of interest is "How large must a problem be in order to achieve some desired performance, measured in flops?" To address this question, we propose the following benchmarking approach: For a given kernel, the vendor should run the problem size of choice, n, that will demonstrate maximum performance, S(n) (flops). Then, for the same kernel, the vendor should determine the smallest possible problem size, $n_2$, such that $S(n_2) = S(n) / 2$. $n_2$ is the classic value that vendors used to provide to indicate how large vectors needed to be before the user observed 1/2 the asymptotic rate. It is still a relevant measure. Note that there need not be a restriction on using the same number of processors when measuring S(n) and $S(n_2)$. From $n_2$, it is possible for scientists to estimate performance on realistic problems, rather than on hero-only computations where the memory on each processor has been saturated in order to hide communication overhead, etc. Machines having large S(n) and small $n_2$ are clearly going to be able to deliver high performance over a broader job mix than those having comparable S(n) but very large $n_2$. For CFD, it would be sensible to report $n_2$ values for the HPC Challenge Benchmarks G-HPL (to test distributed dense matrix-matrix multiplies), a global multidimensional FFT (either 2D or 3D), and a distributed sparse matrix-vector product test (e.g., a point Jacobi iteration and a conjugate gradient iteration, with the latter providing insight to the overhead associated with global vector reductions).

**I/O considerations:** A powerful computer is not just a set of compute resources, but also entails a high-performance I/O system that can feed the computers and store their results. However, in many scientifically important computations, it is not the CPUs, but the storage system, that limits the overall speed. Some large simulations create output that is a converged product, or a statistical average of many trajectories: these do relatively little I/O. It is the simulations that store every time step where I/O becomes a limiting factor: in these cases we expect large, synchronized parallel write operations, where each processor outputs part of a large file.

The processing of data from observations and experiments may also have critical need for I/O, for example processing large quantities of data may have many small files processed by a CPU farm, with files assigned to processors in a static or dynamic method. It may be a data-mining process, where each file goes through pattern-matching or source detection, and a relatively small database of found objects created. The process may also be one of cleaning or reprojection, where the output is the same size as the input. Experimental data may also be processed in a way that is difficult to predict *ab initio*: for example, the data may be stored in directories in the order in which it was taken from the instrument, but a user wants a subset of the total based on a complex query.

In this case, the file system gets a large number of essentially random-access requests into a collection of millions of files.

The benchmarks should include the data-handling capabilities of a proposed new machine—just as we consider the cargo capacity of a new car as well as its top speed. There are many benchmarks of I/O performance for parallel cluster machines. The NSF has already endorsed the SPIObench [1] code suite, a measure of "synchronized" I/O with large numbers of processors running synchronously to read and write files. Another benchmark suite is the Interleaved or Random (IOR) [2] suite from Livermore, although unfortunately the random component is deprecated, and the suite now looks rather like a DOE version of [1] that is from DOD.

An excellent paper on benchmarking parallel file systems [3] includes the sort of tests that would round out a testing of file systems, it measures the effect of multiple simultaneous clients and the rate at which files can be created or opened, stress on the metadata server as well as the bulk data handling.

[1] Streaming Parallel I/O Benchmark
http://www.nsf.gov/pubs/2006/nsf0605/spiobench.tar.gz

[2] The I/O Stress Benchmark Codes
http://www.llnl.gov/asci/purple/benchmarks/limited/ior/

[3] Shared Parallel Filesystems in Heterogeneous Linux Multi-Cluster Environments
http://csc.cs.colorado.edu/~tufo/pubs/tufo-2005-lci.pdf

# Section 6: Description of Proposed Benchmark Codes

## Title: Global Arrays

http://www.emsl.pnl.gov/docs/global/

**Description:** The Global Arrays (GA) toolkit provides an efficient and portable "shared-memory" programming interface for distributed-memory computers. Each process in a MIMD parallel program can asynchronously access logical blocks of physically distributed dense multi-dimensional arrays, without need for explicit cooperation by other processes. Unlike other shared-memory environments, the GA model exposes to the programmer the non-uniform memory access (NUMA) characteristics of the high performance computers and acknowledges that access to a remote portion of the shared data is slower than to the local portion. The locality information for the shared data is available, and a direct access to the local portions of shared data is provided.

Global Arrays have been designed to complement rather than substitute for the message-passing programming model. The programmer is free to use both the shared-memory and message-passing paradigms in the same program, and to take advantage of existing message-passing software libraries. Global Arrays are compatible with the Message Passing Interface (MPI). The Global Arrays toolkit has been in the public domain since 1994. It has been actively supported and employed in many large codes since then, including nearly all scalable chemistry codes.

**Significance:** The increasing complexity of scientific applications and high-performance computers combined with the need to exploit tens of thousands of processors presents the developer of a new code with substantial challenges. The amount of effort required by such a project has lead to a gulf between the innovation in small university groups and the larger projects or groups that can be partly addressed by adoption of tools that provide higher levels of abstraction than conventional message passing and also provide a shared-memory-like environment which facilitates the effective use of the aggregate resources of the entire computer. One-sided access to remote memory is also instrumental in eliminating unnecessary synchronizations and in enabling facile dynamic load balancing.

**Benchmark calculations:** The standard GA installation includes a test code for measuring the performance of the basic put, get and accumulate operations. These should be run across all possible levels of the memory hierarchy or communication fabric.

## Title: MILC

**Description:** The MILC Code is a publicly available software suite for the study of Quantum Chromodynamics (QCD), the theory of the strong interactions. The overwhelming fraction of the allocations in Lattice QCD at NSF and DOE centers currently go to groups making use of the MILC code. Among the topics being studied are:

• the decay properties of strongly interacting particles containing heavy quarks which play a critical role in making precise tests of our current theories of fundamental interactions;

• the properties of strongly interacting matter under extreme conditions, such as those that existed in the early universe and in heavy-ion collision experiments;

• the masses and internal structure of strongly interacting particles.

**Significance:** QCD calculations are very well suited to parallel computers. They take place on regular, four-dimensional lattices. To update degrees of freedom at an individual lattice site or on an individual lattice link, only requires data from a few neighboring sites or links, so most communication is short range. However, a significant number of global sums are made during the course of a calculation. Communications between processors are regular and predictable and can often be overlapped with computation. Despite these simplifying features, considerable effort is required to produce code that is portable and achieves high performance on a wide variety of platforms. This has been done with the MILC code. The code is currently part of the NERSC benchmark suite and is under consideration for the next set of SPEC benchmarks.

In production runs, the bulk of the floating point operations go into the inversion of large sparse matrices. The performance of the inversion routine provides a good indication of the overall performance of the code. However, the MILC code contains timing options for all major phases. In lattice QCD, the physical variables are 3-component complex vectors and 3 by 3 complex matrices. Most computation is either matrix multiplication or matrix applied to vector. This reduces the demands on memory bandwidth from simple binary arithmetic (8 bytes single precision input, 4 bytes single precision output per floating point operation) to 1.45 bytes of input and 0.36 bytes of output per floating point operation for matrix vector products. This demand is much higher than for dense matrix multiplication (LINPACK), and more representative of many scientific codes involving partial differential equations. By running benchmarks with different grid sizes per CPU, we can tune the degree of cache reuse.

The MILC code is also sensitive to interprocessor communication. An $L^4$ hypercube of the 4D lattice is assigned to each processor. If the number of lattice sites in the hypercube is small enough so that it fits entirely into cache, then excellent single processor performance should be obtained. Otherwise, performance is degraded. Cache misses can be minimized by careful layout of the data, and in some cases by pre-fetching. Of course, the fraction of the lattice points that require off-processor data increases as the size of the hypercubes decrease, thereby decreasing the ability to overlap computation and communication, and putting greater strain on the inter-processor communication system. To overlap computation and communication requires an achieved bandwidth (in MB/s) of 0.546 MF/L , where the MF represents the actual computation rate in megaflops/s. These competing effects need to be balanced to obtain optimal performance. It is therefore important to run benchmarks for a range of hypercube sizes, as well as for different numbers of processors.

If the number of lattice sites per processor is kept fixed, then the MILC code scales almost linearly with the number of processors and therefore the size of the problem up to at least 4,096 processors. It is, of course, highly desirable to increase the number of processors for a fixed problem size. A lower limit of $6^4$ lattice sites per processor is more typical.

**Benchmark calculations:** Scaling studies with L = 4, 6, 8, 10, 12 and 14 grids are useful, allowing tests of cache and main memory performance, interprocessor communication. For a machine such a Bluegene with limited memory per CPU and many nodes, the larger values of L may be omitted. Ideally, the study would go the largest number of CPUs available.

## Title: GAMESS

**Description:** The following is extracted from the above URL and the GAMESS summary web page, http://www.msg.ameslab.gov/GAMESS/summary.html.

GAMESS is a program for *ab initio* molecular quantum chemistry. Briefly, GAMESS can compute SCF wavefunctions ranging from RHF, ROHF, UHF, GVB, and MCSCF. Correlation corrections to these SCF wavefunctions include Configuration Interaction, second order Perturbation Theory, and Coupled-Cluster approaches, as well as the Density Functional Theory approximation. Nuclear gradients are available, for automatic geometry optimization, transition state searches, or reaction path following. Computation of the energy hessian permits prediction of vibrational frequencies, with IR or Raman intensities. Solvent effects may be modeled by the discrete Effective Fragment Potentials, or continuum models such as the Polarizable Continuum Model. Numerous relativistic computations are available, including third order Douglas-Kroll scalar corrections, and various spin-orbit coupling options. The Fragment Molecular Orbital method permits use of many of these sophisticated treatments to be used on very large systems, by dividing the computation into small fragments.

A variety of molecular properties, ranging from simple dipole moments to frequency dependent hyperpolarizabilities may be computed. Many basis sets are stored internally, together with effective core potentials, so all elements up to Radon may be included in molecules. Most computations can be performed using direct techniques, or in parallel on appropriate hardware. Graphics programs, particularly the MacMolPlt program for the Macintosh, are available for viewing of the final results.

A detailed description of the program is available in the following journal article: "General Atomic and Molecular Electronic Structure System" M.W. Schmidt, K.K. Baldridge, J.A. Boatz, S.T. Elbert, M.S. Gordon, J.H. Jensen, S. Koseki, N. Matsunaga, K.A. Nguyen, S.Su, T.L.Windus, M. Dupuis, J.A. Montgomery *J. Comput. Chem.*, 14, 1347-1363 (1993).

GAMESS was put together from several existing quantum chemistry programs, particularly HONDO, by the staff of the National Resource for Computation in Chemistry. The NRCC project (1 Oct 77 to 30 Sep 81) was funded by NSF and DOE. It presently comprises roughly 720,000 lines of FORTRAN.

**Significance:** GAMESS (US version) is perhaps the most widely used molecular chemistry program in the world. It is freely available (under a simple license that precludes redistribution) and as such is also a major platform for development of new methods. Although it is neither the most efficient nor the most scalable chemistry code, its extensive functionality, portability, and ready availability makes it the *de facto* choice for chemical benchmarks. GAMESS has substantial parallel capabilities and demonstrated execution on over 1000 processors using the Distributed Data Interface (DDI) which derives from Global Arrays.

**Benchmark Calculations:** The major workhorses of computational chemistry are density functional theory, multi-configuration self-consistent field, second-order perturbation theory, and coupled cluster theory. The most common tasks are to optimize a molecular structure and subsequently compute vibrational frequencies. The suggested benchmarks are DFT (B3LYP) frequency calculation on a sizeable molecule, and a single step in a molecular geometry optimization using spin-unrestricted MP2 in a high-quality basis. Both of these calculations are expected to scale to at least hundreds of processors for adequately sized molecules.

## Title: NAMD

http://www.ks.uiuc.edu/Research/namd/

**Description:** NAMD, recipient of a 2002 Gordon Bell Award, is a parallel molecular dynamics code designed for high-performance simulation of large biomolecular systems. Based on Charm++ parallel objects, NAMD scales to hundreds of processors on high-end parallel platforms and tens of processors on commodity clusters using gigabit ethernet. NAMD is file-compatible with AMBER, CHARMM, and X-PLOR.

James C. Phillips, Rosemary Braun, Wei Wang, James Gumbart, Emad Tajkhorshid, Elizabeth Villa, Christophe Chipot, Robert D. Skeel, Laxmikant Kale, and Klaus Schulten. Scalable molecular dynamics with NAMD. Journal of Computational Chemistry, 26:1781-1802, 2005.

**Significance:** NAMD is the most scalable molecular dynamics program of which we are aware, and has demonstrated execution on over 32K processors of IBM BlueGene/L. Although for few processor simulations there are codes that are somewhat faster for certain benchmarks, NAMD consistently scales better for large processor counts.

NAMD has an extensive range of functionality which has led it to be adopted as a development framework by other groups world wide.

**Benchmark Calculations:** The standard set of benchmarks ([1] and [2]) already developed to compare the performance of NAMD, Amber and CHARMM is a good starting point, but probably too small for testing performance on machines with 10+K processors. The code's authors should be contacted to establish an appropriate test.

[1] http://www.ks.uiuc.edu/Research/namd/performance.html

[2] http://amber.scripps.edu/amber8.bench2.html

## Title: PKDGRAV/Gasoline

http://hpcc.astro.washington.edu/faculty/trq/brandon/
http://130.113.172.122/

**Description**: GASOLINE is used in ground-breaking studies of the structure in the very early Universe, modeling the structure of dark matter surrounding galaxies, and producing one of the first ever realistic simulations of a disk galaxy. So successful is the parallel methodology used in GASOLINE that it has been ported into realms beyond cosmology-sized N-body calculations. It is now routinely exploited to calculate problems in planetesimal formation, planetary disk dynamics, asteroid collisions, and even falling grains of sand in an hourglass. N-body simulations will be crucial in our understanding of cosmological models, the dark matter question, and what made the universe become what we see today.

Astrophysical simulations pose intense challenges for operating in a parallel environment. While many problems in high performance computation are "local" in that the physical effects of a given resolution element are only felt by its immediate neighbors, the force of gravity is long-ranged: everything in a self-gravitating system interacts with everything else. In a gravity simulation, each processor therefore requires data from every other processor in order to compute the solution. Thus, an effective strategy for overcoming interprocessor communication latency is paramount to any efficient implementation.

GASOLINE is a parallel N-body code employing a tree structure which forms the hierarchical representation of the mass distribution. It implements a Lagrangian scheme for the self-gravitating fluid and gas interaction by Smooth Particle hydrodynamics. GASOLINE overcomes the non-locality of gravity by employing a data caching scheme where off-processor data is organized into cache lines, and an off-processor request retrieves not only the requested data itself, but also other data in the same cache line. The tree is then walked in a manner that makes the most efficient use of the data cache. The cache is so effective that typically fewer than 0.01% of requests for off-processor data actually require a message to be sent. GASOLINE can achieve up to 92% linearity on 512 processors. It requires both local and global communications and very long integration times. Communications are asynchronous and can be between any two pairs of processors. Processors that have spatially-adjacent domains in problem-space are more likely to exchange messages. However, the mapping between problem-space topology and network topology is non-trivial: i.e. processors that share a domain boundary in the simulation are rarely physically adjacent to one another. Thus, the communication pattern is essentially random.

GASOLINE's portability, is achieved by encapsulating all interprocessor communication into a Machine Dependent Layer ("MDL," approximately 800 lines of code) that can be rewritten for any parallel architecture. MDLs that have been optimized for nearly every parallel platform that has existed during the last 10 years including MPI, POSIX threads, and Cray Shmem.

**Benchmark calculation:** A possible benchmark would be to follow a pre-galaxy state to its formation.

## Title: Cactus-GR

http://www.cactuscode.org/

**Description:** Cactus is an open source problem solving environment designed for scientists and engineers. Its modular structure easily enables parallel computation across different architectures and collaborative code development between different groups. Cactus originated in the academic research community, where it was developed and used over many years by a large international collaboration of physicists and computational scientists. It has a strong support group, it has been deployed in practically all major HPC environments and has been shown to scale to a large number of processors. Detailed historical timing-information is readily available and is used by a large community.

Within this problem solving environment, particular applications can be built. In particular, applications to solve Einstein equations have been developed which can be employed as a useful benchmark directly related to one of the most important computer simulation project. A new generation of gravitational wave detectors is coming into operation at enough sensitivity to detect gravitational wave signals produced by strongly gravitating astrophysical systems. These waves contain precise information of the source that created them and analyzing the obtained data will provide us with a description of this source. Furthermore, in many cases—like in the collision of black holes—this analysis represents our only venue for understanding a large number of amazing phenomena.

This analysis is to be carried out by confronting the obtained data with theoretical models obtained from Einstein's theory. The highly involved character of the equations describing the theory requires of numerical simulations capable of dealing with the most dynamical stage of the systems (and the one which generates the strongest output in gravitational waves).

The simulation thus requires: 1) the ability to handle 3-4 order difference in length scales (ranging from below the size of the source—the black hole) to the far distances representing the location of a distant observer on earth. 2) to deal with a highly complex PDE system which is non-linear, to evolve about fifty principal variables and that requires thousands of floating point operations per grid point. 3) to be sufficiently accurate that the numerical errors are guaranteed to be smaller than the target physical output, which in itself is small. 4) follow the system for considerably long times. These requirements translate in very high needs in memory and CPU speed which can only (approximately) be reached today by deploying the code in a massive parallel environment and making efficient use of it by incorporating adaptive mesh capabilities in the implementation.

This benchmark is both present, and more importantly, quite forward-looking as simulations of astrophysical systems incorporating General Relativity will represent a significant amount of leading HPC research in the coming decades as long standing questions like the detailed understanding of gamma-ray-bursters; active galactic nuclei, neutron star/black hole collisions, supernovae etc will require incorporating, among other important physical ingredients, General Relativity.

**Benchmark calculation:** For a model HPC hardware one naturally would not expect to deal with a large simulation of a binary black hole system. Sufficient information can be gathered by implementing a benchmark simulating a single black hole system (of size $L_1$=M), placing the computational boundaries at ($L_2$ = 100M) and having a base grid with a resolution of size D=$L_1$/20. and evolving the system for one-crossing time, requiring 2000 time steps.

## Title: PARATEC

http://www.nersc.gov/projects/paratec/

**Description:** Prediction of the properties of real materials and devices from fundamental principles at the quantum level has become a reality on the powerful computers of today. First-principle density Functional Theory (DFT) is a major theoretical framework to study the electronic and structural properties of molecules and solids. The pseudo-potential method in a plane-wave basis set (PP-PW) is the most efficient implementation of DFT to study the properties of a system with periodic boundary conditions: bulk systems, surfaces and one-dimensional structures such as nanotube.

**Significance:** PARATEC (PARAllel Total Energy Code) is a representative of many PP-PW codes that have become common computational tools for a wide spectrum of research communities from quantum

chemistry, condensed matter theory to experimental materials research. PARATEC is "a package designed primarily for a massively parallel computing platform and can run on serial machines". PARATEC employs an efficient parallel 3-dimensional FFT, direct and iterative eigensolvers, and minimization methods. The performance of PARATEC on any HPC system is a reliable benchmark for many PP-PW codes. Furthermore, it is a good performance indicator of other applications. Multi-dimensional FFT and eigensolvers are notoriously difficult problems and are the bottlenecks of many applications due to their scaling with an increasing problem size. For the science enabled by the petascale HPC, their performance on many processors will determine if we can do the science as planned. We propose three classes of input. Here, we assume that the number of processors is one common variable for all the benchmarks.

**Benchmark inputs**:

1. For a fixed-size problem for the entirely computing resources and for a fixed-size problem per processors. A simple example is a bulk system, e.g. Si and the number of Si atoms in a problem can be used as the variable for the benchmark.

2. For a problem with mixed boundary conditions, e.g., surface. In this case, two variables should be used: the number of *k*-points to control the accuracy and the number of particles in a non-periodic directions with a fixed *k*-points in the other periodic directions.

3. For the scaling with respect to the basis size for fixed-size problems.

## Title: FLASH

http://flash.uchicago.edu/website/home/

**Description:** leading edge large numerical simulations must often deal with an insufficient computational power. (This is to ensure the outcome is below the desired error level and/or incorporate further processes in the description of the problem at hand.) Additionally, even when adequate resources are available, a more efficient use of these resources can be achieved by adapting the discretization lengths adopted to what is needed and not more.

An ideal way to deal with these issues is to adapt the mesh employed in the simulation ensuring it enables an accurate description of the solution's features without wasting resources by adopting too fine a mesh. As opposed to the case in linear PDEs where one could *a priori* adjust the mesh from what is expected in the solution, a non-linear problem can develop finer structure not suspected at the outset. Turbulence phenomena is a prime example of this.

Thus, it is necessary to let the solution itself determine what is a suitable mesh to employ for a given situation. This is achieved by adaptively modifying the mesh by monitoring the solution's error (measured by an appropriately defined truncation error evaluator) and adding or discarding points/elements or particles employed in the description of the problem.

Although quite effective AMR schemes have been developed for use in single processors and shared memory machines, efficient schemes over a distributed computational environment are still subject of significant research at the theoretical and practical levels. Here, a challenge is to decide where to spawn new grids (either locally at a given processor which owns the portion of the grid to be refined) or globally among several nodes whose load might not be as severe. A similar issue arises when grids which are no longer needed, have to be discarded leading to load-inbalance. An efficient scheme, then requires good algorithms to deal with these issues, and the ability to communicate large amounts of data over all processors involved.

**Significance:** Many of today's (and future) leading HPC problems absolutely require adaptive mesh refinement, not only to achieve a reasonable description with otherwise inadequate computational resources but also to expedite achieving the solution in problems that would be well resolved with a fixed and even uniform mesh.

We thus find it imperative to suggest benchmarks implementing some sort of adaptive re-gridding technique. A good candidate for this is the Eulerian AMR FLASHcode which implements AMR through the PARAMESH driver. The code is being used for astrophysical applications by a large number of scientists, it is highly portable and has a well recognized support group behind it. There is ample historical data on scalability over a large number of platforms.

# Appendix A: Letter to Participants of the Meeting

September 20, 2005

Dear Colleagues:

Thank you for agreeing to participate in the Town Hall Meeting on October 17, 2005. This meeting builds on the work of the Mathematical and Physical Sciences (MPS) Cyberscience Workshop convened at NSF in April 2004, to address cyberscience and the cyberinfrastructure required to realize the opportunities cyberscience presents. The town hall meeting will focus in greater depth on high performance computing (HPC), one aspect of cyberinfrastructure, and specifically on how we can measure in a meaningful way the performance of high performance computing systems on the problems of the mathematical and physical sciences. HPC performance measures are important for many uses within the community and for NSF where good performance indicators can guide the process to procure high performance computing systems.

The science that will be done using HPC systems and an understanding of the HPC systems, codes, and other resources that will be needed to do the science are the starting points in determining meaningful performance metrics. It is hoped that this meeting will identify candidate benchmark codes that show promise for being solid measures of HPC system performance. It may be useful for discussions to note that performance is not limited to processor speed; there are many factors that contribute to the time to solution. The discussion process will likely uncover many issues and we appreciate that a comprehensive response may not be possible by the end of the October 17, meeting. So, the October 17, meeting should be regarded as a progress report where key issues and some candidate benchmark codes will be identified. It is part of a longer process that will culminate in a future meeting, one that may take place possibly as early as December or January. A tentative meeting agenda is posted on the web pages. The participants will set the final agenda of the meeting; it will be a result of interactions and discussions prior to the meeting primarily through the web forum.

Minimizing the barriers to participation is a key consideration. The Materials Computation Center (MCC) has agreed to host the meeting and Duane Johnson will serve as the Chair. The physical location of the meeting will be at the NCSA Building in Access Grid Room 3000 on the campus of the University of Illinois at Urbana-Champaign. Participation is also possible through other Access Grid (AG) nodes and teleconference. NSF will support travel to UIUC or to a participating AG node that is closest to you. Information on travel to UIUC and how to reserve a room at the Hampton Inn is available on the meeting web pages (http://www.mcc.uiuc.edu/nsfhpc). Because of the many access node possibilities, there are no specially reserved hotel rooms for AG nodes other than the AG node at Urbana. Travel arrangements to UIUC or an appropriate access node should be made on your own. You will be reimbursed for your travel expenses through the MCC; please keep all your receipts. A form, available at http://www.mcc.uiuc.edu/travel/travel_reimbursement_form.pdf, is needed to claim expenses. For those who are unable to travel on this date, telephone access will be provided on the day of the meeting.

To facilitate communication among participants prior to the meeting, there is an interactive (wiki style) website available through a link from the main meeting pages. Comments can be posted to existing discussion threads and new discussion threads can be created as needed. At some point in a thread, it may be that the web forum does not provide an appropriate or efficient way to discuss a particular idea. Feel free to establish contact by other means (e.g. telephone, e-mail, etc.) using contact information for participants available through the weblink on the web forum page. Please post an appropriate comment in the web forum on the resolution of such discussions to preserve intellectual coherency and to inform the rest of the group.

This letter contains a username / password combination that will enable you to access the registration pages and to join the web forum. Please register as soon as possible. Registration is required to be reimbursed for your expenses. If you are at an access node, please reserve your local access node (http://www.accessgrid.org/) for the NSF Computing Meeting on October 17, 10:00 AM - 7:30 PM.

HPC resources are used in diverse ways across MPS. The number of participants in any meeting is limited. You are encouraged to consult with your colleagues across the mathematical and physical sciences. Note that this is particularly important at AG nodes where there are usually many researchers who use HPC resources. Your local colleagues are encouraged to take part on the meeting day as space and resources permit.

Feel free to contact us with questions at any time. The web forum also has a thread dedicated to communication with NSF. Divisional points of contact and their contact information appear below and on the meeting website. Duane Johnson (duanej@uiuc.edu; 217-265-0319) can help with questions related to the Materials Computation Center and its role in support of the meeting. For questions regarding logistics at the Urbana meeting and procedures for travel reimbursement, contact Ramona Simpson, Materials Computation Center at UIUC (rlsimpso@uiuc.edu, 217-333-1371).

Thank you again for participating in this important meeting, particularly on short notice.

Best regards,

Daryl W. Hess, Division of Materials Research, dhess@nsf.gov, (703) 292-4942

Nigel Sharp, Division of Astronomical Sciences, nsharp@nsf.gov, (703) 292-4905

Celeste Rohlfing, Chemistry Division, crohlfin@nsf.gov, (703) 292-4962

Leland Jameson, Division of Mathematical Sciences, ljameseon@nsf.gov, (703) 292-4883

Barry Schneider, Physics Division, bschneid@nsf.gov, (703) 292-7383

# Appendix B: List of Town Hall Meeting Participants

Below is the name, affiliation, email address, and area of expertise for each of the 32 participants.

**Bruce Allen (University of Wisconsin - Milwaukee)**
ballen@uwm.edu; http://www.lsc-group.phys.uwm.edu/
Area of specialization: We analyze data from gravitational wave detectors as part of the LIGO Scientific Collaboration. We also run the Einstein@Home distributed computing project, which is currently getting around 25 Tflops round the clock.

**Mihai Anitescu (Argonne National Laboratory)**
anitescu@mcs.anl.gov
Area of specialization: optimization

**Julian Bunn (CalTech)**
Julian.Bunn@caltech.edu
Area of specialization: High performance computing and networking for Particle Physics.

**David Ceperley (University of Illinois Urbana-Champaign)**
ceperley@uiuc.edu
Area of specialization: Simulation of quantum systems, condensed matter physics, low temperature physics, quantum Monte Carlo methods

**Andrew Connolly (University of Pittsburgh)**
ajc@phyast.pitt.edu
Area of specialization: Analysis of data from large astronomical surveys (e.g. the SDSS) to understand galaxy formation and cosmology. Typical applications range from clustering statistics (correlation function, density estimation, parameter estimation and model fitting), image analysis, and tracking of moving sources. Typical analyses are data mining applications (including grid based) with some parallel implementations.

**Qiang Cui (University of Wisconsin)**
cui@chem.wisc.edu
Area of specialization: Computational and theoretical chemistry; Hybrid QM/MM simulations of condensed phase, especially biological, systems; multi-scale simulations of biological processes that involve energy transduction.

**Thom Dunning (NCSA, University of Illinois Urbana-Champaign)**
tdunning@ncsa.uiuc.edu
Area of specialization: Electronic structure of molecules and quantum chemistry with an emphasis on highly correlated methods (multireference configuration interaction and coupled cluster methods). Areas of interest include molecular structure, energetics, and reactivity.

**Paul Fischer (Argonne Lab)**
fischer@mcs.anl.gov
Area of specialization: Computational fluid dynamics, with a particular focus on high-order numerical methods, large-scale parallel algorithms, and scalable iterative solvers, for simulations of incompressible flow, magnetohydrodynamics, and electromagnetics. Application areas include vascular flows, heat transfer in reactor cores, gravity currents, spatiotemporal chaos, combustion, and nanophotonics.

**Mark Gordon (Ames Lab)**
mark@si.fi.ameslab.gov
Area of specialization: We develop the electronic structure code GAMESS (General Atomic and Molecular Electronic Structure System), a general-purpose electronic structure code. We focus on highly parallel codes for applications in chemistry, biochemistry, and materials science.

**Steven Gottlieb (Indiana U.)**
sg@indiana.edu
Area of specialization: Computational physics, in particular, Lattice Gauge Theory a numerical approach to the strong interaction, on of the four forces of Nature. Knowledgeable about parallel computing, benchmarking, performance analysis and tuning of the MILC codes for lattice QCD.

**Robert Harrison (U. of Tennessee, ORNL)**
robert.harrison@utk.edu
Area of specialization: Theoretical and computational chemistry, high-performance computing, fast numerical algorithms, scalable algorithms.

**Daryl Hess (National Science Foundation)**
dhess@nsf.gov
Area of specialization: NSF Program Director, Division of Materials Research, Condensed Matter and Materials Theory Program.

**Leland Jameson (National Science Foundation)**
ljameson@nsf.gov
Area of specialization: Hyperbolic conservation laws.

**Mark Jarrell (University of Cincinnati)**
jarrell@physics.uc.edu
Area of specialization: Strongly correlated many-body electronic and magnetic systems. Information theory and Maximum Entropy Methods.

**Duane Johnson (University of Illinois Urbana-Champaign)**
duanej@uiuc.edu
Area of specialization: Computational materials science with emphasis on electronic-structure and thermodynamics of materials, especially alloys with disorder. Algorithm development, e.g. O(N) KKR-based methods and multiscaling via genetic-programming or molecular dynamics embedded in Space-Time Discontinuous Galerkin FEM. Applications to systems with disorder and defects, surfaces, supported nano-assemblies.

**Jeongnim Kim (NCSA, University of Illinois Urbana-Champaign)**
jnkim@uiuc.edu
Area of specialization: study the electronic structures of complex systems and development of materials simulation codes.

**Luis Lehner (Louisiana State University)**
lehner@lsu.edu
Area of specialization: General Relativity, Numerical Relativity, Computational Physics

**Todd Martinez (University of Illinois Urbana-Champaign)**
tjm@spawn.scs.uiuc.edu
Area of specialization: Theoretical and computational chemistry, electronic-structure theory, and quantum and classical dynamics.

**Bill McCurdy (U. California at Davis, LBL)**
cwmccurdy@ucdavis.edu
Area of specialization: Area of research: theoretical atomic and molecular physics, employing large scale computations based on grid methods for problems (like molecular double photoionization) with two electrons in the continuum.

**A. Alan Middleton (Syracuse University)**
aam@syr.edu
Area of specialization: Condensed matter physics and computational physics, with a focus on statistical physics, soft condensed matter, and random magnets: the dynamics and static behaviors of many-degree-of-freedom heterogeneous systems. I use direct numberical integration of large sets of coupled ODE's, combinatorial optimization, and Monte Carlo.

**Paul Ricker (University of Illinois at Urbana-Champaign)**
pmricker@uiuc.edu; http://www.astro.uiuc.edu/~pmricker
Area of specialization: My research discipline is computational astrophysics and cosmology, including evolution of galaxies and clusters of galaxies, globular clusters, the interstellar medium, and supernova explosions. I am also interested in computer science topics bearing on scientific computing, particularly architectures for parallel hydrodynamic codes used in astrophysical simulation.

**Celeste Rohlfing (National Science Foundation)**
crohlfin@nsf.gov
Area of specialization: Program Director, NSF Division of Chemistry

**Thomas Russell (National Science Foundation)**
trussell@nsf.gov
Area of specialization: Program director for applied math and computational math in NSF/MPS/DMS. Additional responsibilities include cyberscience/cyberinfrastructure. Research on numerical methods for PDEs, scientific computing, applications to fluid flows in subsurface porous media.

**Barry Schneider (National Science Foundation)**
bschneid@nsf.gov
Area of specialization: Theoretical AMO, Computational Physics

**Nigel Sharp (National Science Foundation)**
nsharp@nsf.gov
Area of specialization: Program Director, Extragalactic Astronomy and Cosmology, plus Math.Innovations, Cyberinfrastructure, and all CI-related issues.

**Robert Sugar (University of California, Santa Barbara)**
sugar@physics.ucsb.edu, http://gabriel.physics.ucsb.edu/~sugar/
Area of specialization: Study of quantum chromodynamics (QCD), the theory of the strong interactions of subatomic physics

**Bruce Taggart (National Science Foundation)**
gtaggart@nsf.gov
Area of specialization: Program Director for Condensed Matter and Materials Theory

**Priya Vashishta (U. of Southern California)**
priyav@usc.edu
Area of specialization: Multimillion Atom Simulation of Materials Molecular Dynamics Simulations Fast Multipole Methods, Bond Order Potentials, Qeq schemes, ReaxFF interactions representing different class of complexities in computing, communications, network latency and bandwidth.

**Gregory A. Voth (University of Utah)**
voth@chem.utah.edu, http://www.cbms.utah.edu/
Area of specialization: Molecular dynamics simulation of membranes, filaments, proton transport & solvation. Development of multiscale modeling methodologies.

**Tim Warburton (Rice University)**
timwar@rice.edu
Area of specialization: High-order discontinuous Galerkin methods for Maxwell's equations. Artificial radiation boundary conditions for the time domain wave equation. Software and algorithm development for high order finite element methods.

**Thomas Weber (National Science Foundation)**
tweber@nsf.gov
Area of specialization: Materials Research

**Roy Williams (CalTech)**
roy@cacr.caltech.edu
Area of specialization: Data-intensive computing in astronomy, especially. I am co-director of the NSF National Virtual Observatory, which is developing service-oriented architecture to allow astronomers to publish, discover, and utilize important data.